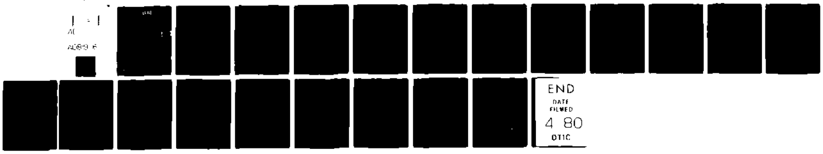
AD-A081 926 ILLINOIS UNIV AT URBANA-CHAMPAIGN COORDINATED SCIENCE LAB F/G 9/2
EXPERIENCE WITH THE EVALUATION OF NATURAL LANGUAGE QUESTION ANS--ETC(U)
JAN 79 H TENNANT N00014-75-C-0612
UNCLASSIFIED WP-18 NL

1 of 1
AD
A081926



END
DATE
FILMED
4 80
DTIC

9 WORKING PAPER 18

6 Experience with the Evaluation of Natural Language Question Answerers.

10 Harry Tennant
Advanced Automation Group
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Urbana, IL 61801

11 January 1979

12 21

14 WP-18

Abstract

Research in natural language processing could be facilitated by thorough and critical evaluations of natural language systems. Two measurements, conceptual and linguistic completeness, are defined and discussed in this paper. Testing done on two natural language question answerers demonstrated that the conceptual coverage of such systems should be extended to better satisfy the needs and expectations of users. Three heuristics are presented that describe how conceptual coverage of question answerers should be extended.

Key words and phrases: natural language processing, question answerers, evaluation, completeness, coverage.

This work was supported by the Office of Naval Research under Contract 15 N00014-75-C-0612

Working papers are informal papers intended for internal use.

Forty-two papers related to natural language processing were presented at the International Joint Conference on Artificial Intelligence in 1977. Of these, only four articles made any attempt to evaluate the work being presented other than by giving a few examples of correctly analyzed language. Only one paper [Mann, Moore, Levin, 1977] dealt with the problem of assessing the performance of a model of knowledge understanding. The authors had developed a model for recognizing context shifts in human dialog, and tested it by comparing the performance of the model to human judgements on the same text. None of the papers that described implemented understanding programs reported on their performance as language understanders. In the other 38 papers, there were no examples of language that was handled inappropriately. The reader has very little hope of thoroughly understanding the capabilities of the systems and techniques described. Nor is there any objective way for a reader to compare different approaches to the problems of natural language processing when the approaches are described the way they were in the IJCAI proceedings.

The papers from that conference are not unusual. The accepted practice of performance description in natural language does not include objective evaluation of the work or thorough description of limitations. The primary technique in current use for demonstrating achievements is to provide a list of about twenty examples of correctly analyzed user inputs. In general, there are no negative results given. There is no mention made as to where the results came from: e.g., from actual users or members of the development team. This type of description suggests that the reader should extrapolate from the examples to understand the full capabilities of the system. This is an uncertain process, at best!

The lack of evaluation of natural language processing research leave several crucial questions about the work unanswered. Readers are unsure as to what concepts are included in the system, what accomodations have been made for language variations between users, the restrictions on the discourse domain or database, the restrictions on data manipulation capabilities, and the restrictions on inferencing capabilities. There is usually no information about the match between the facilities included in the system and the actual needs of the users. In addition, there is little information on what kind of performance would be required of a natural language processor to allow users to carry out tasks at various levels of complexity. This last deficit precludes any hope of a succinct description of the state of the art. There is no clear way to state where the research stands relative to its goals or what the goals will accomplish if they are met. This situation fosters confusion about the progress and achievements of natural language processing research both inside and outside the research community. Petrick writes:

> "Proponents of natural language systems cite the success of prototype systems and claim the time is ripe to construct large practical natural language systems. However, those who oppose such systems claim that our current knowledge cannot support an undertaking of such difficulty. A perusal of the natural language question-answering literature indicates the reason for these contradictory claims...with the single exception of the [LUNAR] system there have been no attempts to evaluate the capacity of a natural language question-answering system to satisfy the needs of the user community for which the system was designed. Furthermore, there have been few attempts to characterize the extent of the syntactic and semantic coverage of English provided." [Petrick, 1976]

The lack of evaluation impedes the development of the techniques of natural language processing by leaving readers uncertain about what has been accomplished as opposed to what has been speculated. McDermott writes:

> "This muddle finally hurts those following in the researcher's path. Long after he has his PhD or tenure, inquiring students will be put off by the document he has left behind. He seems to have solved everything already, so the report says, yet there is no tangible evidence of it besides the report itself. No one really

> wants to take up the problem again, even though the original research is a partial success or even a failure! If a student decides [the researcher's idea] is a good idea, and wants to study it, people will assume he is 'merely implementing' an already fully designed program."[McDermott, 1976]

The solution to these problems must lie in more thorough and critical descriptions of systems.

A natural language processing program could be evaluated along several dimensions. One could consider factors such as speed, memory requirements, or the clarity of algorithms and interpretation rules. But the most critical dimension is the linguistic and conceptual performance of the system: what concepts can it understand and how free are users to express those concepts in the ways they find most convenient. In our work on evaluation we have concentrated on these two aspects of performance. We have further restricted ourselves to natural language question answering systems. In the paragraphs that follow, we will describe performance goals and measurement in more detail, then discuss the findings to which their use has led us.

1.0 PERFORMANCE GOALS AND MEASUREMENT

A natural language processor can be seen from two points of view: that of the designer and that of the user. The designer examines the domain of discourse of a particular application and maps out the range of concepts that are within it. The range of concepts that are built into the natural language system by its designer is the CONCEPTUAL COVERAGE of the system. The designer must also provide for the various ways in which the user will generate his statements and requests to refer to the concepts covered in the system. The set of features or range of linguistic phenomena that have been built into the system to allow for the diversity of users' language is the LINGUISTIC

COVERAGE of the system.

The difference between conceptual coverage and linguistic coverage is fairly distinct. Say a user had a particular aircraft in mind that he wanted to refer to. Any of the following references could be appropriate in the proper context:

plane 3
serial number 3
the plane with serial number 3
plane number 3
the plane
it
she
the Skyhawk
the other one
the last one I mentioned
the repaired one

The diversity in the form of the reference may be described as syntactic, semantic, pragmatic, or lexical. The means by which the phrases are interpreted as references to that one particular plane are some of the elements that constitute the linguistic coverage of the system. The concept of the plane is one of the elements of the conceptual coverage of the system. Other elements of conceptual coverage that are indicated by some of the phrases are that the plane has a serial number, 3, that it is a Skyhawk type aircraft, that it has been referred to previously in the dialog and that it has been repaired.

Noun phrases are not the only units that presuppose elements of conceptual coverage. Prepositions can presuppose concepts (e.g., above, behind, inside) as can comparative constructions (e.g., greater than, more flights than). Conjunctions can imply time sequence, set membership, or causality, all members of the conceptual coverage of a system. There are others, this list is representative, not exhaustive.

When a natural language system is designed, the designers build a certain conceptual coverage and linguistic coverage into it. The measure of their success is how well the needs of the users of the system have been anticipated.

Users see a natural language question answering system and the database to which it interfaces through the perspective of their own needs and habits. When a particular user approaches a database question answerer he will expect it to include certain concepts, and he will form his utterances in his accustomed way. The degree to which the concepts that are expected by a set of users can actually be found in the system's conceptual coverage is the CONCEPTUAL COMPLETENESS of the natural language processor, with respect to the set of users. Similarly, the degree to which the language of a set of users is appropriately analyzed by the system is the LINGUISTIC COMPLETENESS of the natural language processor with respect to that set of users.

Conceptual completeness, as defined here, is similar to but differs from the definition of completeness given in [Woods, Kaplan and Nash-Webber, 1972]. They defined a system as "logically complete if there is a way to express any request which it is logically possible to answer from the database". Defining conceptual completeness in terms of conceptual coverage has several advantages. First, it extends the range of concepts from those included in the database to the entire domain of discourse. Second, it permits the restriction of the definition from all requests that are logically possible, to all that a set of users find useful. Third, it retains the requirement that there must be at least one way to reference the concepts.

The definition of linguistic completeness is intended to be similar to the definition of fluency in [Woods, Kaplan and Nash-Webber, 1972]. Both are measures of the variety of ways in which a concept covered by a system can be expressed. The definition presented here differs from theirs in that it is defined with respect to a set of users. If a measure of linguistic completeness were not made relative to a set of users, it must be made relative to some universal scale or standard taxonomy of linguistic phenomena. Such cannot be readily provided and widely accepted.

On the other hand, defining conceptual completeness with respect to the database and linguistic completeness with respect to some universal scale of language is appealling. Defining conceptual completeness with respect to the database allows one to develop an algorithm to determine what the conceptual coverage of a system must be to make it complete. Unfortunately, a natural language system that is complete in this sense tends to leave users unsatisfied, as will be described in more detail below. Similarly, linguistic completeness based on a universal scale of some kind allows one to develop an algorithm for determining the elements of linguistic coverage of a natural language system. Making conceptual completeness and linguistic completeness measures with respect to a set of users leaves the designer without algorithms, only with heuristics. Algorithms may be more aestheticly appealling than heuristics, but we feel that heuristics can do a better job of describing the reality of performance of natural language processors.

It is interesting to consider the current description techniques for natural language processors in terms of conceptual and linguistic coverage, and conceptual and linguistic completeness. When a paper describing a natural language system presents twenty or so questions that are appropriately analyzed, the questions include concepts from the system's conceptual coverage, and their forms and features suggest elements of the system's linguistic coverage. Unfortunately, the conceptual and linguistic coverage of the system is not fully specified by what is found in the examples, and one cannot generalize from them. If no claim is made that the examples were in some sense typical of user inputs, nothing can be inferred about conceptual completeness or linguistic completeness. If the paper goes on to explicitly mention phenomena such as ellipsis, pronoun reference, or comparatives, a comment is being made about the elements of linguistic coverage (and, in the case of comparatives, the conceptual coverage of the concept of comparison). Since this is not being related to the needs of a set of users, it says nothing about linguistic completeness. One might imagine how these elements of linguistic coverage might affect linguistic completeness. However, more must be learned about the language people use when interacting with computers before natural language systems can be engineered to meet specified goals for linguistic and conceptual completeness within an acceptable tolerance for an expected set of users.

2.0 PERFORMANCE TESTING

Performance testing for conceptual linguistic completeness was conducted on two natural language question answering systems, PLANES and the Automatic Advisor. PLANES [Waltz and Goodman, 1977] [Waltz, 1978] is a large and powerful natural language question answerer that interfaces the user to a relational database of naval aircraft flight and maintenance records. The Automatic Advisor [Tennant, 1977] is a much smaller and simpler system that interfaces the user to information about engineering courses offered at a university. The testing that was done of these systems was not intended to produce a thorough evaluation of either system. Instead, it was done to exercise our theories on how evaluations of natural language processors might be conducted.

The test users were university students who were familiar with neither the natural language processors nor their databases. PLANES users were given a description of the domain of discourse that consisted of about 600 words of text to familiarize them with some of the nomenclature used and the kind of information in the database. Users of the Automatic Advisor were given no more than a couple of sentences stating the domain of discourse. The nomenclature and probable database contents were assumed to be familiar to university students.

After this short familiarization, the users were given a set of database problems to solve using the natural language systems. There were two methods of problem generation. The first was to ask users to generate problems for other users based on the domain description. They had been given no experience on the system, so they had no knowledge of the kinds of problems the system could actually handle. The second was that those of us who were familiar with the two systems designed problems. The second set was generated because users had great difficulties in solving problems from the first set. The reasons for these difficulties is one of our most interesting findings, and will be discussed below. As users attempted to solve their database problems with the systems, they were given assistance only under two conditions: 1) a catastrophic system failure or 2) a high frustration level. Sessions usually lasted about two hours.

The database problem composers were instructed to specify the problems in such a way as to minimally affect the user's language. To this end, problems were described with partially completed histograms, tables, and graphs and users were asked to complete them, or with verbose or elaborate descriptions that users would want to paraphrase, or as high level problems requiring decomposition by the users.

One could test the general competence of the natural language system by simply giving database problems to users and letting them attempt to solve them. Conceptual completeness can be tested by giving the database problems to someone who is very familiar with the system, like the system designers. If a problem can be solved at all the conceptual coverage must include the concepts indicated by the problem. Linguistic completeness can be tested by giving users the problems that were solved in the conceptual completeness test. If a user utterance cannot be handled appropriately by the system, it is probably due to a limitation of linguistic coverage. It could, however, be due to a limitation of the conceptual coverage that was not discovered in the conceptual completeness test. This is why tests for conceptual completeness and linguistic completeness are not entirely independent.

3.0 TEST RESULTS

The most interesting findings were in the area of conceptual completeness. Papers describing natural language processors frequently concentrate on linguistic coverage with the implicit assumption that the database determines the range of conceptual coverage. We had made this assumption ourselves in the design of PLANES and the Automatic Advisor and in the database problems we composed. However, we found this orientation to be askew from the reality of the situation. This is exactly the reason that users were less able to solve the problems composed by other users, but were more successful with the problems composed by the members of our research group. The casual users were approaching the question answering system with the activities that the database described in mind. They composed their problems that way and asked their questions that way. Our mistake in the architecture of the natural language processors and the reason that the problems we composed were easier to solve on the systems was that we were familiar with the database contents, and thought in those terms. The activities orientation is quite reasonable and evidently the one that should be supported for casual users. This suggests heuristic 1 for natural language question answerer designers:

(1) Conceptual coverage should be considered from the point of view of real world activities, events and objects, rather than the point of view of a database which supposedly describes them.

Heuristic 1 should be helpful for identifying a broad class of erroneous presuppositions that may be embedded in user's queries. In the question "Which planes that flew on March 2, 1970 had maintenance on March 3?", one presupposition is that there were in fact some planes that flew on March 2. If the database lists the planes that flew on particular days, this would be a Kaplan-type presupposition [Kaplan, 1978]. It is possible to identify violated presuppositions of this type by checking for null returns to queries that are predicated on each presupposition. However, if there were no information on flights in the database, it would be a more general form of presupposition that Kaplan's techniques could not handle. Codd [Codd, et al, 1978] has made a first attempt at identifying this broader form of presupposition by recognizing keywords that suggest concepts that are not within the database (although Kaplan-type presupposition is not handled). Heuristic 1 should guide designers toward expanding their views of what users presume is included in the system's conceptual coverage.

Another result relating to conceptual completeness was evidence against the common assumption that users' utterances are limited to database queries. Most natural language question answerers rely on the assumption that the user will ask a question, the system will query the database to answer it, then the user will ask another and so on. This is not the way conversations work. In testing PLANES and the Automatic Advisor, users frequently made utterances that were not intended to be interpretted as database queries. Various examples follow.

1. About the database:
    1. "What do you know"?
    2. "What planes do you know about?" (a query could be done to answer this, but, in PLANES, it would be extremely expensive)
    3. "Tell me about f4s".
    4. "Is that everything you know about math 195?"
2. About vocabulary:
    1. "What is a buser?"
    2. "What does howmal mean?"
3. Context setting:
    1. "Now I am talking about the year 1970".

2. "Year is 1970".

3. "1 am interested in a7's".

4. Reference to discourse objects:

   1. "What was the last plane I talked about?"

   2. "Now combine the top" (meaning combine two tables that were the answers to the last two questions into one table).

   3. "The other ones I just mentioned" (attempt to correct the system after it found an erroneous antecedent for a pronoun).

5. Verifying or summarizing results:

   1. "Then infe 210 and infe 211 must be taken at the same time".

   2. "So all I need is junior standing and math 195".

6. Multiple query utterances:

   1. "What parts failed and what were the parts removed or installed".

   2. "How many flights did plane 48 have in dec 1969, and did it have over 50 nor hours?"

   3. "List plane 48 flights and flight hours in december 1969. Were there more than 50 hours not operationally ready?"

7. Multiple utterance queries:

   1. "How many aircraft flew more than 10 flights in 1973? in 1972? in 1970? List aircraft by number of flights for 1970."

   2. "Which parts failed? removed? installed?"

8. Miscellaneous:

   1. "This is a test"

   2. "What time is it?"

   3. "Eat a bag of [obscenity deleted]"

   4. "What's going on here?"

Each of these examples was actually input by users using one of the natural language processors. Many more examples in each of these categories could be imagined. A number of additional examples can be found in Malhotra's work [Malhotra, 1975] on a hypothetical management information system. One category that appeared in his dialogs consisted of questions about the linquistic and data manipulation capabilities of the system. He also found that users attempted to extend the capabilities of the system by defining new terms. Examples from Malhotra are as follows.

1. About capabilities:
   1. "Can you calculate percentages?"
   2. "How are contribution margins calculated?"
   3. "Is transportation cost included in overhead?"
   4. "Do you perform mathematical computations."
   5. "List the functions you can perform."

2. Extentions:
   1. "Compute profit for 1972 and 1973 according to the following formula: actual unit sales by product times list price minus production cost for the product summed over all products less overhead cost for the year."
   2. "Call last displayed quantity sales growth."
   3. "Let alloc be ((overhead/production cost)*total production cost) for each product."

These examples show clearly that question answerers cannot operate on a one-database-query-per-sentence basis. This is stated in the following heuristic and its corollaries.

(2) A natural language question answerer must not be limited to interpreting each user utterance as a database query.

Corollaries: A question answerer should expect the following types of utterances in addition to single question database queries:
(2a) utterances about the database structure and contents
(2b) utterances about the system's vocabulary
(2c) utterances that set the context
(2d) references to objects of the discourse
(2e) utterances meant to verify or summarize results
(2f) utterances requiring more than one database query
(2g) queries composed of more than one sentence
(2h) references to the capabilities of the system
(2i) utterances intended to extend the conceptual coverage of the system

The inspiration for a third heuristic came while testing the Automatic Advisor. It pertains to pragmatic relationships between elements of a database domain. It depicts one of the ways in which the characteristics of the database can affect the task of engineering natural language interfaces and it has important implications for "portable" natural language systems.

One of the data domains of the Automatic Advisor is that of the topics that are covered by the engineering courses in the database. The topics covered in the lower level courses tended to be more general, those in the upper level courses were more specific. The Automatic Advisor included no pragmatic relationships among the elements of the topics domain. Consequently, when asked for the courses that dealt with computers, it returned one course, Digital Systems, the only one which explicitly mentioned "computers" as a topic. The courses that it failed to return due to a lack of pragmatic relationsips among data elements included three on digital hardware and computer architecture, one on computer vision, two on graphics, two on artificial intelligence, one on automata theory and one on real time signal processing. This list doesn't mention a host of other courses that make extensive use of computers.

Misleading answers, whether incomplete or totally erroneous, are the most insidious kind that a natural language processor can make. The presence of an answer inspires the user's confidence that he has been understood. When his confidence is betrayed by a misleading answer, he may simply go blitnely on unaware that he is working with bad data.

The problem of misleading answers due to missing pragmatic relationships is not pecular to the Automatic Advisor, but is common to "portable" natural language processors. Harris states that "[the portability of ROBOT] is largely possible because of the way ROBOT makes use of the database as an existing semantic structure" [Harris, 1977]. Indeed, ROBCT, the Automatic Advisor, and other systems are evidence that much semantic and pragmatic information does lie in the structure of the database, but the tests with the Automatic Advisor have illustrated the limitations of relying solely on the database for this information. These findings lead to heuristic 3.

(3) Conceptual coverage should include the pragmatic relationships between data elements even though these relationships may not be expressed in the database.

4.0 LINGUISTIC COMPLETENESS

Much work has been done and discussed in the literature on identifying the elements of linguistic coverage for improved linguistic completeness. While our user testing has not led to the discovery of new elements of linguistic coverage, it can be usefully applied to evaluate the effectiveness of linguistic feature handling programs. Statistics of usage from user testing can also indicate the relative importance of features to a class of users.

User testing has several advantages that makes it an appealling way to measure linguistic completeness. First, user testing relates linguistic completeness measurements to a set of users, the advantages of which were discussed above. Second, the design decisions that are made in one part of a natural language program generally have a strong effect on other parts of the system. This suggests that the evaluation of performance, whether for conceptual or linguistic completeness, must consider the natural language processor as a whole. If subsystems are considered separately, there is little hope of having the evaluation results for one system to carry over to another. One system, for example, may have a separate syntactic parser, while another system may embed syntatic constraints in a semantic grammar. An evaluation of the syntactic parser would be no more useful to the design of one as an evaluation of the semantic grammar would be to the other. If one were interested in ascertaining the relative abilities of the two systems to handle various syntactic variations of a sentence, it would be studied most directly at the user level.

There are several disadvantages to user testing for linguistic completeness, however. First, users are adaptable. If a system is found to be unable to handle a particular construction, a user may simply avoid using that construction. If the user diagnoses the problem quickly, he would adapt his language to that accepted by the system. User adaptation could show the system to be relatively fluent, but force users into expressing themselves in unaccustomed ways. Second, user testing for linguistic completeness is a "weakest link" test. If a natural language system has one poorly designed or poorly implemented component, the linguistic completeness of the entire system will suffer. This technique alone is incapable of identifying which components are responsible for errors. Also, if one were interested in measuring linguistic completeness with respect to linguistic features, the user testing paradigm would need to be altered. One might be interested in, say, the handling of anaphoric definite noun phases, but such phrases may occur infrequently in the users' language; in such a case, the user could be directed to generate lists of questions including the desired feature. Directing a user to use certain constructions for measuring linguistic completeness over linguistic features will affect his spontaniety but may, nevertheless yield useful results. At the current state of development of natural language processing, we expect to learn a great deal through this form of testing.

5.0 CONCLUSION

There is a clear need for natural language processors to be evaluated. The measures and measurement techniques described in this paper are an attempt to show that meaningful testing can be performed and that its results can promote the engineering of better systems.

The heuristics for more complete conceptual coverage, and the test results that prompted them, suggest that in addition to the study of the elements of linguistic coverage, attention should be focussed on conceptual coverage. Without extending conceptual coverage beyond the limits of the database contents, a natural language question answerer can do little more than a formal query language. What's worse is that the natural language version would be much more expensive to run. ●ould also give an initial illusion of intelligence that may leave the user quite disappointed when he learns that it is so conceptually incomplete from his point of view. One mainstay argument in favor of natural language processors over formal query languages is that the use of natural language processors can be learned quickly and recalled easily after a period of disuse. These claims have not been convincingly substantiated, however.

The LADDER system [Sacerodoti, 1977] based on the LIFER parser [Hendrix, 1978] was evaluated [Miller, Hershman, and Kelly, 1978] as a query language, and users were able to use the system with some facility after an hour and a half of instruction specifically on the use of the natural language component (they were selected to be familiar with the activities described by the database). It was conceded that the users would have had little hope of success without instruction. But tests of formal query languages have shown similar results [Reisner, Boyce and Chamberlin, 1975].

It seems clear that the role that natural language question answerers should play is in providing the user with more complete conceptual conceptual coverage than he can have using a formal query language. The user has gained little if he avoids the rigors of learning a formal query language only to be subjected to the rigors of learning the details of the structure of the database. If he must know the minutia of either, he will not be able to use the system effectively on an occasional basis.

References

Codd, E. F., R. S. Arnold, J-M Cadiou, C. L. Chang, N. Roussopoulos. "RENDEZVOUS Version 1: An Experimental English-Language Query Formulation System for Casual Users fo Relational Data Bases". IBM Report RJ2144, 1978.

Harris, L. R. "User Oriented Data Base Query With the ROBOT Natural Language Query System." International Journal of Man-Machine Studies. Vol. 9, 1977, pp. 697-713.

Hendrix, G. G., E. D. Sacerodoti, D. Sagalowicz, J. Slocum. "Developing a Natural Language Interface to Complex Data." ACM Transactions on Database Systems, 1978.

Kaplan, S. J. "On the Difference Between Natural Language and High Level Query Languages." Proceedings of ACM78. Washington, D. C., 1978.

Malhotra, A. "Design Criteria for a Knowledge-Based English Language System for Management: An Experimental Analysis." Ph.D. Thesis, MIT, MAC TR-146, 1975.

Mann, W. C., J. A. Moore, J. A. Levin "A Comprehension Model for Human Dialog" Proceedings of the International Conferences on Artificial Intelligence 1977, p. 77.

McDermott, D. "Artificial Intelligence Meets Natural Stupidity" in <u>SIGART Newsletter</u>, April, 1976, p. 4.

Miller, H. G., R. L. Hershman, R. T. Kelly. "Performance of a Natural Language Query System in a Simulated Command Control Environment."

Naval Electronics Systems Command, 1978.

IBM Journal of Research and Development, July, 1976, p. 314.

Reisner, P., R. F. Boyce, D. D. Chamberlin. "Human Factors Evaluation of Two Data Base Query Languages--Square and Sequel." Proceedings of National Computer Conference, 1975, pp.447-452.

Sacerodoti, E. D. "Language Access to Distributed Data With Error Recovery." Proceedings of the International Joint Conferences on Artificial Intelligence, 1977, pp. 196-202.

Tennant, H. "The Automatic Advisor." Master's Thesis, University of Illinois at Chicago Circle. 1977.

Waltz, D. L., B. A. Goodman. "Writing a Natural Language Data Base System." Proceedings of the International Joint Conferences on Artificial Intelligence, 1977, pp. 144-150.

Waltz, D. L. "An English Language Question Answering System for a Large Relational Database." CACM, vol. 21, July, 1978, pp. 526-539.